COMMUNICATION SYSTEMS LAB 7 DATE-12/10/2021

NAME – JAGRIT LODHA

ID – 2019A3PS0165P

SECTION – P4

[I am submitting late because I was travelling to Pilani yesterday]

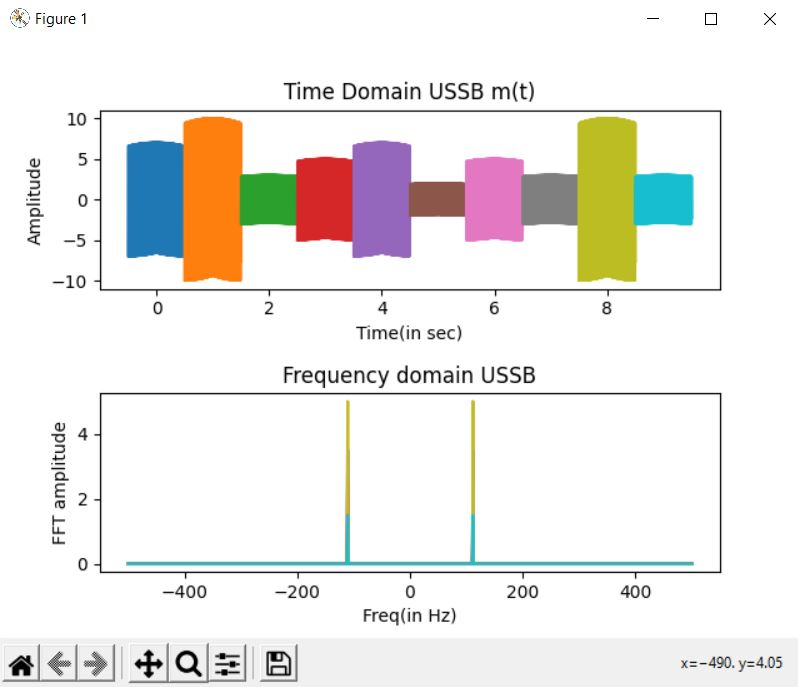
PYTHON

TASK 1.1 –

CODE:

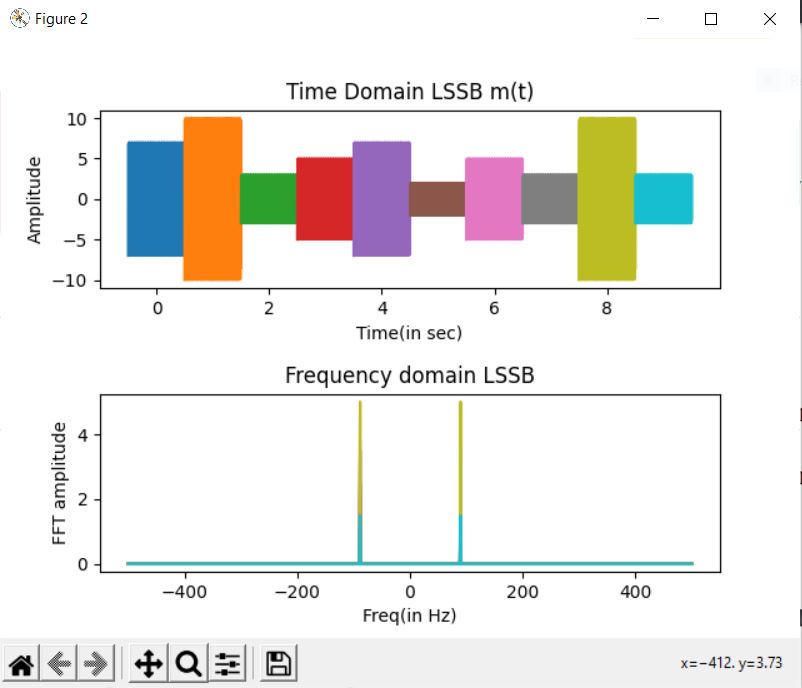
import numpy as np  
import matplotlib.pyplot as plt  
from numpy.fft import fft  
import random  
from scipy.signal import hilbert  
  
N = 11 # ID No - 2019A3PS0165P  
signal\_duration = 10  
fc = 100  
fs = 10\*fc  
  
for T in range(signal\_duration):  
 t\_signal = np.arange(-0.5,0.5,1/fs)  
 Am = random.randint(1,10)  
 m\_t = Am\*np.cos(2\*np.pi\*N\*t\_signal)  
 c\_t = np.cos(2\*np.pi\*fc\*t\_signal)  
 USSB = m\_t\*c\_t - np.imag(hilbert(m\_t))\*np.sin(2\*np.pi\*fc\*t\_signal)  
 LSSB = m\_t\*c\_t + np.imag(hilbert(m\_t))\*np.sin(2\*np.pi\*fc\*t\_signal)  
  
 plt.figure(1)  
 plt.subplot(2,1,1)  
 plt.plot(t\_signal+T, USSB)  
 plt.title('Time Domain USSB m(t)')  
 plt.xlabel('Time(in sec)')  
 plt.ylabel('Amplitude')  
  
 yf1 = fft(USSB) / fs  
 N1 = len(yf1)  
 yf\_abs\_sorted1 = np.fft.fftshift(abs(yf1))  
 freq\_axis = np.linspace(-fs / 2, fs / 2, N1)  
 plt.subplot(2,1,2)  
 plt.title('Frequency domain USSB')  
 plt.xlabel('Freq(in Hz)')  
 plt.ylabel('FFT amplitude')  
 plt.plot(freq\_axis, yf\_abs\_sorted1)  
 # plt.pause(0.1)  
  
 plt.figure(2)  
 plt.subplot(2,1,1)  
 plt.plot(t\_signal+T, LSSB)   
 plt.title('Time Domain LSSB m(t)')  
 plt.xlabel('Time(in sec)')  
 plt.ylabel('Amplitude')  
  
 yf2 = fft(LSSB) / fs  
 N2 = len(yf2)  
 yf\_abs\_sorted2 = np.fft.fftshift(abs(yf2))  
 plt.subplot(2,1,2)  
 plt.title('Frequency domain LSSB')  
 plt.xlabel('Freq(in Hz)')  
 plt.ylabel('FFT amplitude')  
 plt.plot(freq\_axis, yf\_abs\_sorted2)  
 # plt.pause(0.1)  
  
plt.show()

**Time and frequency plots for Upper SSB** –



NOTE – The frequency peaks for Upper SSB are at (fc+N) (111) and (-fc-N) (-111), which can be verified from the above plots.

**Time and frequency domain plots for Lower SSB** –



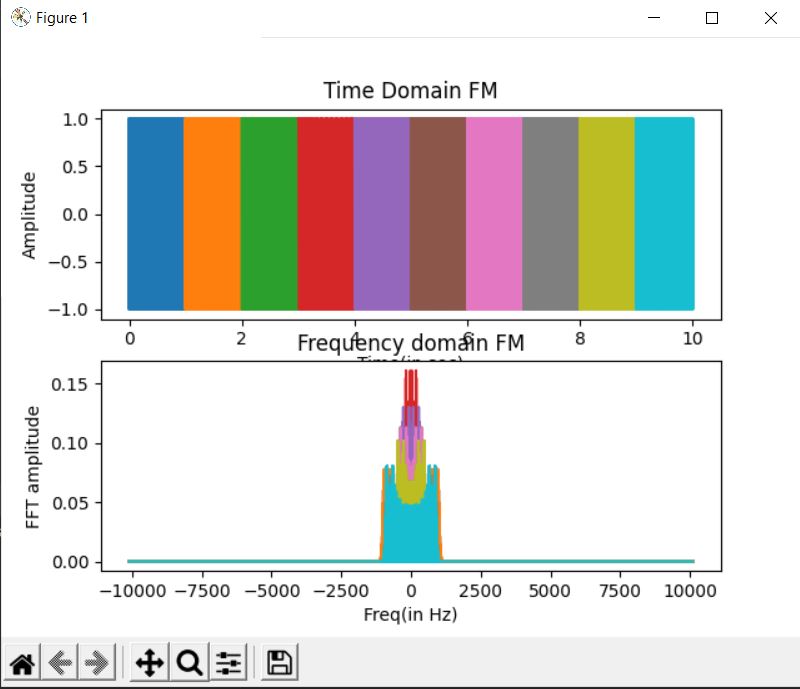
NOTE – The frequency peaks for Upper SSB are at (fc-N) (89) and (-fc+N) (-89), which can be verified from the above plots.

TASK 1.2 -

CODE:

import numpy as np  
import matplotlib.pyplot as plt  
from numpy.fft import fft  
import random  
  
N = 11  
signal\_duration = 10  
fc = 100  
kf = 100  
B = 2\*(kf\*10 + N)  
fs = 10\*B  
  
for T in range(signal\_duration):  
 t\_signal = np.arange(-0.5,0.5,1/fs)  
 Am = random.randint(1,10)  
 m\_t = Am\*np.cos(2\*np.pi\*N\*t\_signal)  
 FM = np.cos(2\*np.pi\*fc\*t\_signal + (Am\*kf/N)\*np.sin(2\*np.pi\*N\*t\_signal))  
 plt.figure(1)  
 plt.subplot(2,1,1)  
 plt.plot(t\_signal+0.5+T, FM)  
 plt.title('Time Domain FM')  
 plt.xlabel('Time(in sec)')  
 plt.ylabel('Amplitude')  
 yf = fft(FM) / fs  
 N1 = len(yf)  
 yf\_abs\_sorted = np.fft.fftshift(abs(yf))  
 freq\_axis = np.linspace(-fs / 2, fs / 2, N1)  
 plt.subplot(2,1,2)  
 plt.title('Frequency domain FM')  
 plt.xlabel('Freq(in Hz)')  
 plt.ylabel('FFT amplitude')  
 plt.plot(freq\_axis, yf\_abs\_sorted)  
 plt.pause(0.1)  
plt.show()

**Time and frequency plot for FM signal** -



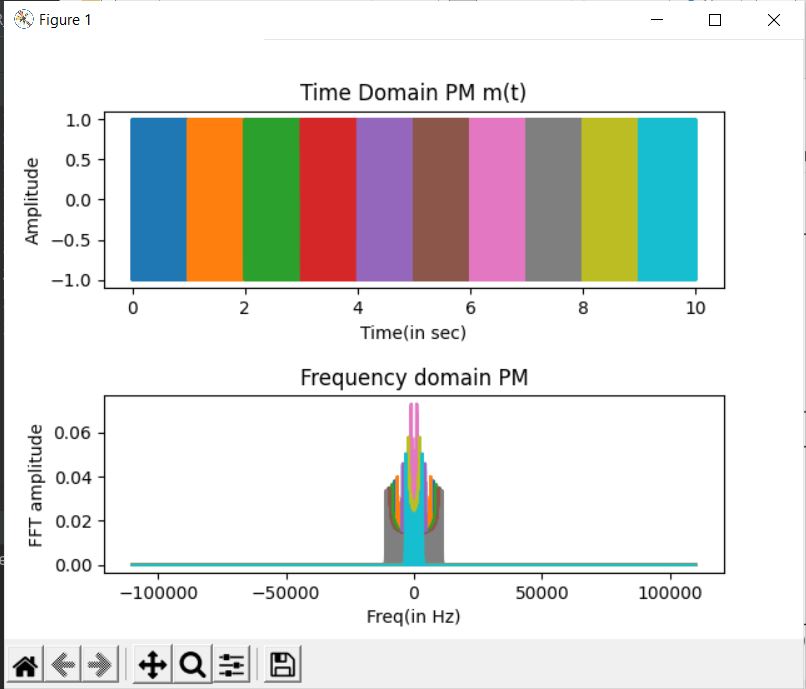
We can clearly observe the frequency modulated plots in the above figure.

TASK 1.3 –

CODE:

import numpy as np  
import matplotlib.pyplot as plt  
from numpy.fft import fft  
import random  
  
N = 11  
signal\_duration = 10  
fc = 100  
kp = 100  
B = 2\*(N + kp\*10\*N)  
fs = 10\*B  
  
for T in range(signal\_duration):  
 t\_signal = np.arange(-0.5,0.5,1/fs)  
 Am = random.randint(1,10)  
 m\_t = Am\*np.cos(2\*np.pi\*N\*t\_signal)  
 PM = np.cos(2\*np.pi\*fc\*t\_signal + kp\*m\_t)  
 plt.figure(1)  
 plt.subplot(2,1,1)  
 plt.plot(t\_signal+0.5+T, PM)  
 plt.title('Time Domain PM m(t)')  
 plt.xlabel('Time(in sec)')  
 plt.ylabel('Amplitude')  
 yf = fft(PM) / fs  
 N1 = len(yf)  
 yf\_abs\_sorted = np.fft.fftshift(abs(yf))  
 freq\_axis = np.linspace(-fs / 2, fs / 2, N1)  
 plt.subplot(2,1,2)  
 plt.title('Frequency domain PM')  
 plt.xlabel('Freq(in Hz)')  
 plt.ylabel('FFT amplitude')  
 plt.plot(freq\_axis, yf\_abs\_sorted)  
 # plt.pause(0.1)  
plt.show()

**Time and frequency plot for PM signal** -



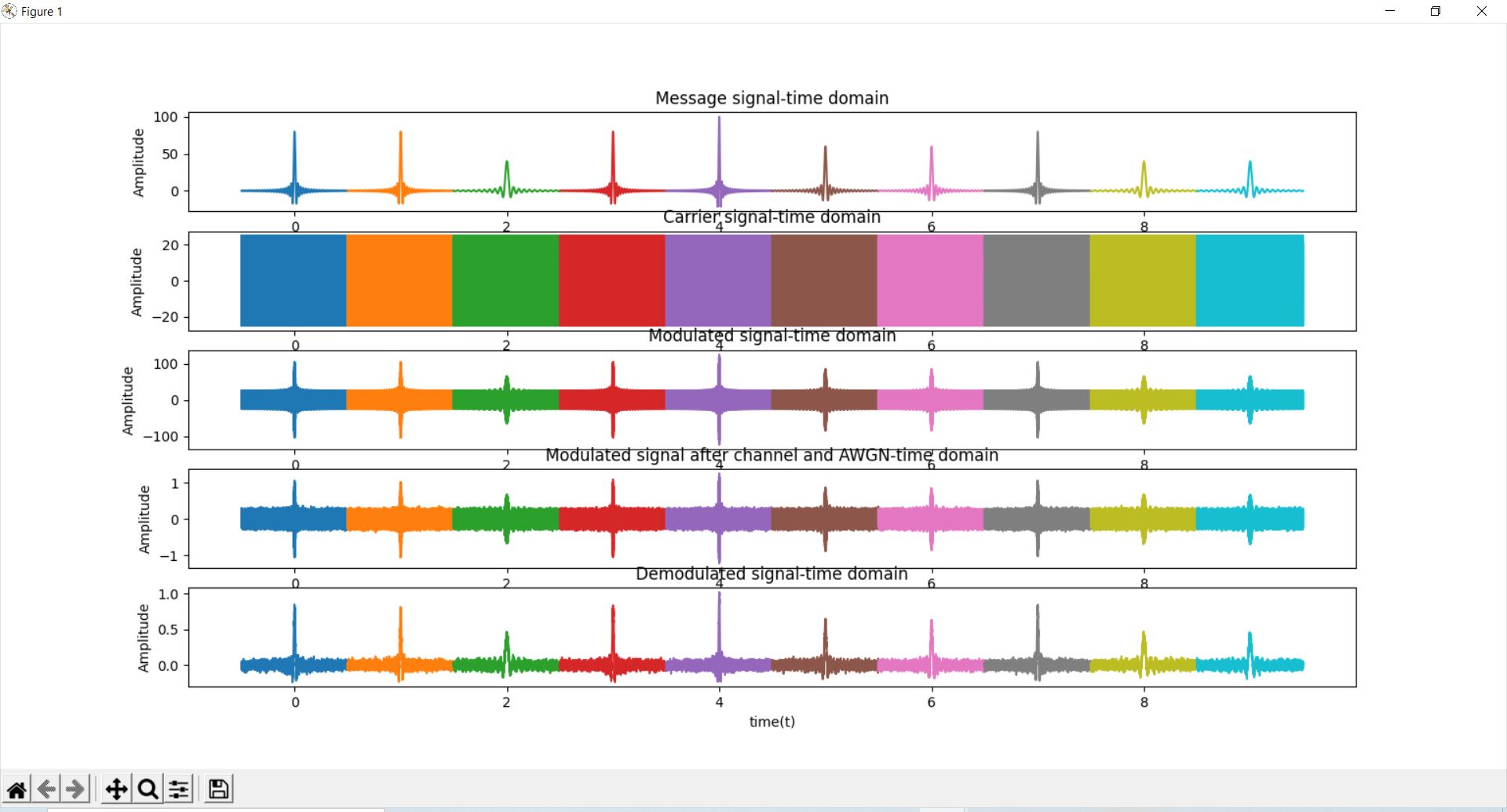
NOTE – We can clearly observe the phase modulated plots in the above figure.

TASK 2 –

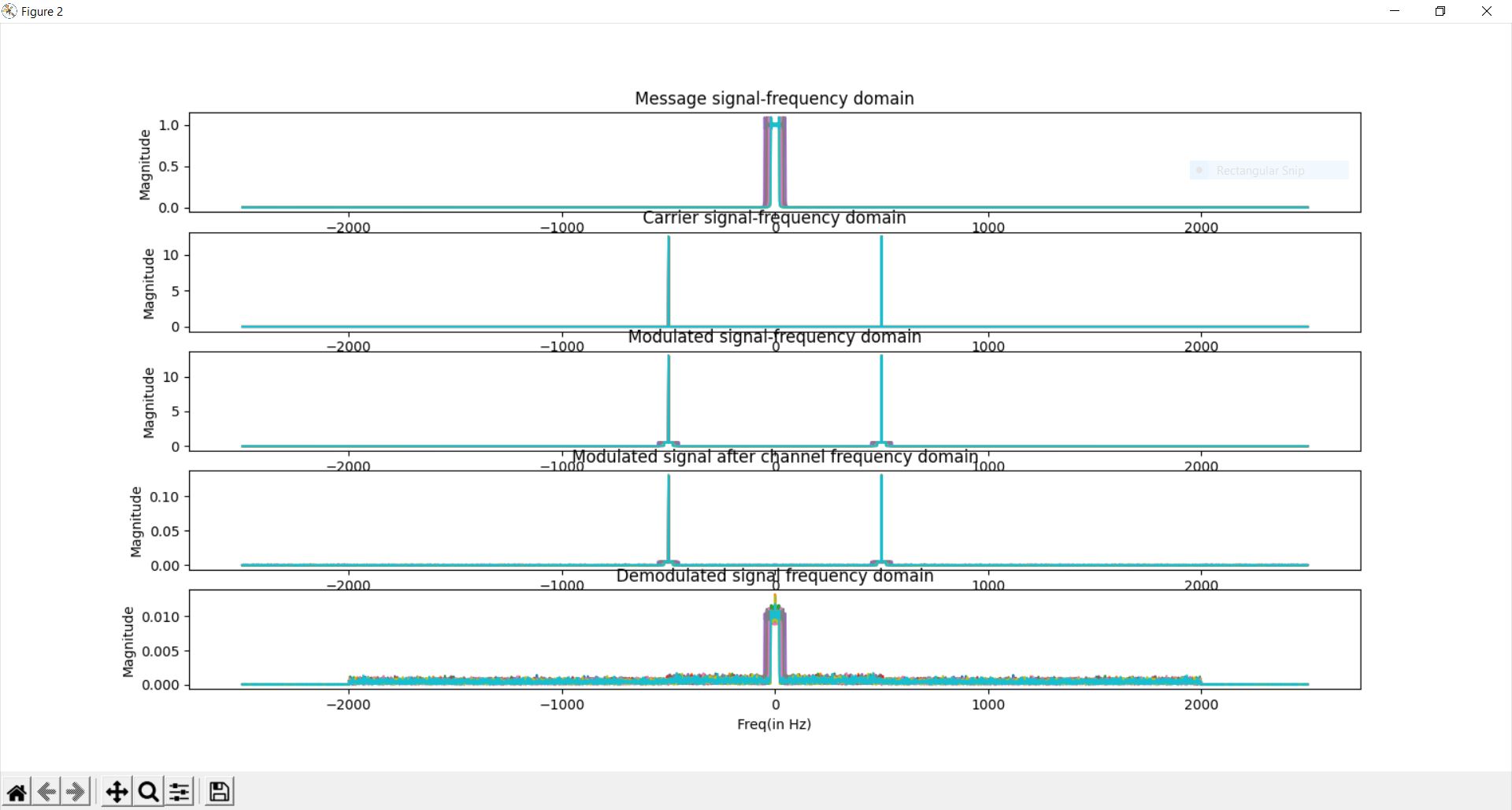
CODE:

import numpy as np  
import matplotlib.pyplot as plt  
import random  
from scipy.signal import hilbert  
  
signal\_duration = 10  
for T in range(10):  
 t\_start = -0.5  
 t\_stop = 0.5  
 carrier\_freq = 500  
 fs = 10\*carrier\_freq  
 ts = 1/fs  
 time = np.arange(t\_start,t\_stop,ts)  
 A = 25  
 carrier\_t = A\*np.cos(2\*np.pi\*carrier\_freq\*time)  
 carrier\_f = np.fft.fftshift(abs(np.fft.fft(carrier\_t)/fs))  
 len\_time = len(time)  
 freq\_axis = np.linspace(-fs/2,fs/2,len\_time)  
 B = random.randint(1,5)  
 m\_t = 20\*B\*np.sinc(20\*B\*time)  
 m\_f = np.fft.fftshift(abs(np.fft.fft(m\_t)/fs))  
 mod\_t = (1+m\_t/A)\*carrier\_t  
 mod\_f = np.fft.fftshift(abs(np.fft.fft(mod\_t)/fs))  
 mu = 0  
 sigma\_sq = 0.001  
 sigma = np.sqrt(sigma\_sq)  
 n\_t = mu+sigma\*np.random.randn(len(time))  
 op\_t = 0.01\*mod\_t+n\_t  
 op\_f = np.fft.fftshift(abs(np.fft.fft(op\_t)/fs))  
 op\_demod\_t = abs(hilbert(op\_t))-A\*0.01  
 op\_demod\_f = np.fft.fftshift(abs(np.fft.fft(op\_demod\_t)/fs))  
 plt.figure(1)  
 plt.subplot(5,1,1)  
 plt.plot(time+T,m\_t)  
 plt.title('Message signal-time domain')  
 plt.xlabel('time(in sec)')  
 plt.ylabel('Amplitude')  
 plt.figure(1)  
 plt.subplot(5, 1, 2)  
 plt.plot(time + T, carrier\_t)  
 plt.title('Carrier signal-time domain')  
 plt.xlabel('time(in sec)')  
 plt.ylabel('Amplitude')  
 plt.figure(1)  
 plt.subplot(5, 1, 3)  
 plt.plot(time + T, mod\_t)  
 plt.title('Modulated signal-time domain')  
 plt.xlabel('time(in sec)')  
 plt.ylabel('Amplitude')  
 plt.figure(1)  
 plt.subplot(5, 1, 4)  
 plt.plot(T+time, op\_t)  
 plt.title('Modulated signal after channel and AWGN-time domain')  
 plt.xlabel('time(in sec)')  
 plt.ylabel('Amplitude')  
 plt.figure(1)  
 plt.subplot(5, 1, 5)  
 plt.plot(time + T, op\_demod\_t)  
 plt.title('Demodulated signal-time domain')  
 plt.xlabel('time(t)')  
 plt.ylabel('Amplitude')  
 plt.figure(2)  
 plt.subplot(5, 1, 1)  
 plt.plot(freq\_axis, m\_f)  
 plt.title('Message signal-frequency domain')  
 plt.xlabel('Frequency (in Hz)')  
 plt.ylabel('Magnitude')  
 plt.figure(2)  
 plt.subplot(5, 1, 2)  
 plt.plot(freq\_axis, carrier\_f)  
 plt.title('Carrier signal-frequency domain')  
 plt.xlabel('Freq(in Hz)')  
 plt.ylabel('Magnitude')  
 plt.figure(2)  
 plt.subplot(5, 1, 3)  
 plt.plot(freq\_axis, mod\_f)  
 plt.title('Modulated signal-frequency domain')  
 plt.xlabel('Freq(in Hz)')  
 plt.ylabel('Magnitude')  
 plt.figure(2)  
 plt.subplot(5, 1, 4)  
 plt.plot(freq\_axis, op\_f)  
 plt.title('Modulated signal after channel frequency domain')  
 plt.xlabel('Freq(in Hz)')  
 plt.ylabel('Magnitude')  
 plt.figure(2)  
 plt.subplot(5, 1, 5)  
 plt.plot(freq\_axis, op\_demod\_f)  
 plt.title('Demodulated signal frequency domain')  
 plt.xlabel('Freq(in Hz)')  
 plt.ylabel('Magnitude')  
 # plt.pause(0.5)  
plt.show()

The plots obtained in time domain –



The plots obtained in frequency domain –



NOTE –

* We can see that the transmitted modulated signal is being demodulated and we almost get back the original message signal.
* There is a little distortion in the received signal due to the additive white gaussian noise.
* The time axis runs from -0.5 to 9.5 in order to show the sinc pulse in both its positive and negative side.